CSE 2021 Computer Organization

Appendix Part 1

The Basics of Logic Design

Outline

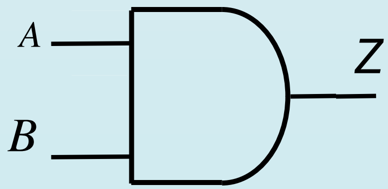
- Fundamental Boolean operations
- Deriving logic expressions from truth tables
- Boolean Identities
- Simplifying logic expressions using Boolean identities
- Combinational and sequential circuits

Boolean Algebra

- Boolean algebra is the basic math used in digital circuits and computers.
- A Boolean variable takes on only 2 values: {0,1} , {T,F}, {Yes, No}, etc.
- There are 3 fundamental Boolean operations:
  - AND, OR, NOT

Fundamental Boolean Operations

AND

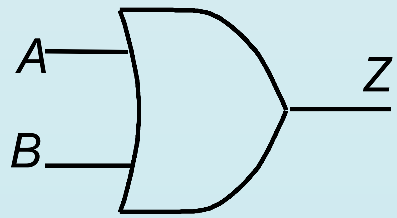


Z=A*B (AB)

Truth Table

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

OR

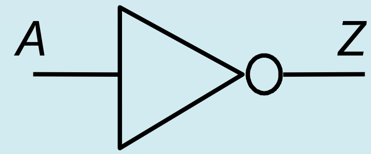


Z=A+B

Truth Table

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

NOT



Z=$\bar{A}$

Truth Table

A	Z
0	1
1	0

Boolean Algebra

- A *truth table* specifies output signal logic values for every possible combination of input signal logic values
- In evaluating Boolean expressions, the *Operation Hierarchy* is: 1) NOT 2) AND 3) OR. *Order can be superseded using (…)*
- *Example:* $A = T, B = F, C = T, D = T$
- What is the value of $Z = (\overline{A} + B) \cdot (C + \overline{B} \cdot D)$?

$$Z = (\overline{T} + F) \cdot (C + \overline{B} \cdot D) = (F + F) \cdot (C + \overline{B} \cdot D)$$

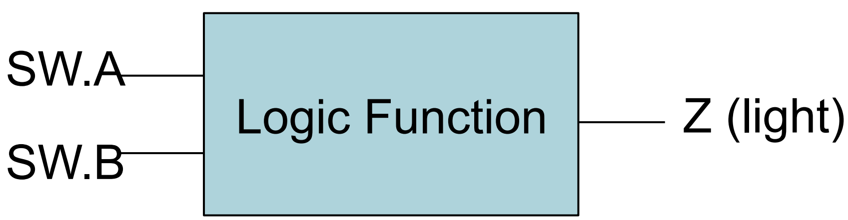
$$= F \cdot (C + \overline{B} \cdot D) = F$$

Deriving Logic Expressions From Truth Tables

Light must be ON when both switches A and B are OFF, or when both of them are ON.

Truth Table:

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	1



- *What is the Boolean expression for Z?*

Minterms and Maxterms

- Minterms
  - AND term of all input variables.
  - For variables with value 0, apply complements
- Maxterms
  - OR factor with all input variables
  - For variables with value 1, apply complements

A	B	Z	Minterms	Maxterms
0	0	1	$\bar{A}.\bar{B}$	$A+B$
0	1	0	$\bar{A}.B$	$A+\bar{B}$
1	0	0	$A.\bar{B}$	$\bar{A}+B$
1	1	1	AB	$\bar{A}+\bar{B}$

Minterms and Maxterms

- A function with n variables has $\mathbf{2^n}$ minterms (and Maxterms) – exactly equal to the number of rows in truth table
- Each minterm is true for exactly one combination of inputs
- Each Maxterm is false for exactly one combination of inputs

A	B	Z	Minterms	Maxterms
0	0	1	$\bar{A}.\bar{B}$	$A + B$
0	1	0	$\bar{A}.B$	$A + \bar{B}$
1	0	0	$A.\bar{B}$	$\bar{A} + B$
1	1	1	AB	$\bar{A} + \bar{B}$

Equivalent Logic Expressions

- Two <u>equivalent</u> logic expressions can be derived from Truth Tables:

1. *Sum-of-Products* (SOP) expressions:
   - Several AND terms OR'd together, e.g.

$$A\overline{B}C + \overline{A}B\overline{C} + ABC$$

2. *Product-of-Sum* (POS) expressions:
   - Several OR terms AND'd together, e.g.

$$(\overline{A} + \overline{B} + C)(A + B + \overline{C})$$

Rules for Deriving SOP Expressions

1. Find each row in TT for which output is 1 (rows 1 & 4)
2. For those rows write a minterm of all input variables.
3. OR together all minterms found in (2):

Such an expression is called a *Canonical* SOP

A	B	Z	Minterms	Maxterms
0	0	1	$\bar{A}.\bar{B}$	$A+B$
0	1	0	$\bar{A}.B$	$A+\bar{B}$
1	0	0	$A.\bar{B}$	$\bar{A}+B$
1	1	1	AB	$\bar{A}+\bar{B}$

$$Z = \overline{A}\,\overline{B} + AB$$

Rules for Deriving POS Expressions

1. Find each row in TT for which output is 0 (rows 2 & 3)
2. For those rows write a maxterm
3. AND together all maxterm found in (2): Such an expression is called a *Canonical* POS.

A	B	Z	Minterms	Maxterms
0	0	1	$\bar{A}.\bar{B}$	$A+B$
0	1	$\boxed{0}$	$\bar{A}.B$	$\boxed{A+\bar{B}}$
1	0	$\boxed{0}$	$A.\bar{B}$	$\boxed{\bar{A}+B}$
1	1	1	AB	$\bar{A}+\bar{B}$

$$Z = (A+\overline{B})(\overline{A}+B)$$

CSOP and CPOS

- Canonical SOP: $Z = \overline{A}\,\overline{B} + AB$
- Canonical POS: $Z = (A + \overline{B})(\overline{A} + B)$
- Since they represent the same truth table, they should be identical

Verify that $Z = \overline{A}\,\overline{B} + AB \equiv (A + \overline{B})(\overline{A} + B)$

- *CPOS and CSOP expressions for the same TT are logically equivalent. Both represent the same information.*

Activity 1

Derive SOP and POS expressions for the following TT.

A	B	Carry
0	0	0
0	1	0
1	0	0
1	1	1

Activity 1

Derive SOP and POS expressions for the following TT.

A	B	Carry	Minterms	Maxterms
0	0	0	A'B'	A+B
0	1	0	A'B	A+B'
1	0	0	AB'	A'+B
1	1	1	AB	A'+B'

SOP: Carry=AB

POS: Carry=(A+B)(A+B')(A'+B)

Boolean Identities

Boolean Identities

- Useful for simplifying logic equations.

	(a)	(b)
1	$\overline{\overline{A}} = A$	$\overline{\overline{A}} = A$
2	$A + \text{false} = A \quad (A + 0 = A)$	$A \cdot \text{true} = A \quad (A \cdot 1 = A)$
3	$A + \text{true} = \text{true} \quad (A + 1 = 1)$	$A \cdot \text{false} = \text{false} \quad (A \cdot 0 = 0)$
4	$A + A = A$	$A \cdot A = A$
5	$A + \overline{A} = \text{true} \quad (A + \overline{A} = 1)$	$A \cdot \overline{A} = \text{false} \quad (A \cdot \overline{A} = 0)$
6	$A + B = B + A$	$A \cdot B = B \cdot A$
7	$A + B + C = (A + B) + C = A + (B + C)$	$A \cdot B \cdot C = (A \cdot B) \cdot C = A \cdot (B \cdot C)$
8	$A \cdot (B + C) = A \cdot B + A \cdot C$	$A + B \cdot C = (A + B)(A + C)$
9	$\overline{A + B} = \overline{A} \cdot \overline{B}$	$\overline{A \cdot B} = \overline{A} + \overline{B}$
10	$A \cdot B + A \cdot \overline{B} = A$	$(A + B)(A + \overline{B}) = A$
11	$A + A \cdot B = A$	$A(A + B) = A$
12	$A(\overline{A} + B) = A \cdot B$	$A + \overline{A} \cdot B = A + B$
13	$A \cdot B + \overline{A} \cdot C + B \cdot C = A \cdot B + \overline{A} \cdot C$	$(A + B)(\overline{A} + C)(B + C) = (A + B)(\overline{A} + C)$

Duals

Boolean Identities

Identities	Property
1-5	Single variable, foundations of Boolean manipulation
6	Commutative
7	Associative
8	Distributive
9	De Morgan’s
10	Combining
11	Absorption
13	Consensus

Boolean Identities

- The right side is the dual of the left side
  1. Duals formed by replacing

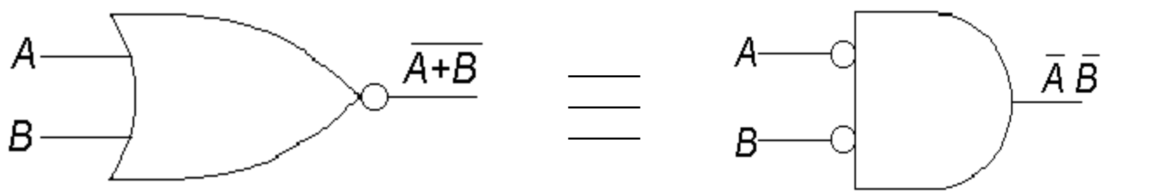
     AND → OR
     OR → AND
     0 → 1
     1 → 0

  2. The dual of any true statement in Boolean algebra is also a true statement.

Boolean Identities

- DeMorgan's laws very useful: 9a and 9b

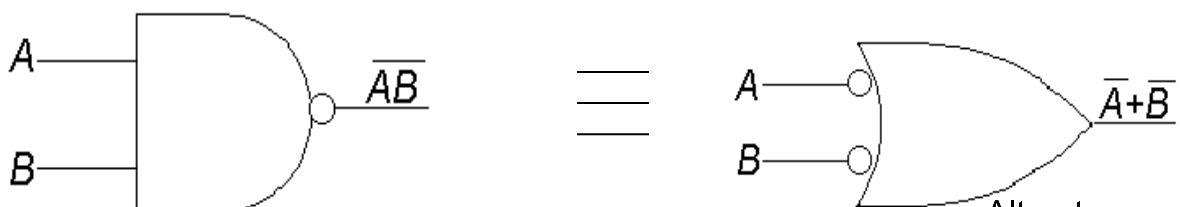
$$\overline{A+B} = \overline{A}.\overline{B}$$



NOR gate

Alt gate rep.

$$\overline{AB} = \overline{A} + \overline{B}$$



Alt gate rep.

NAND gate

Activity 2

Proofs of some Identities:

12b: $$\mathrm{A} + \overline{\mathrm{A}}\mathrm{B} = \mathrm{A} + \mathrm{B}$$

13a: $$AB + \overline{A}C + BC = AB + \overline{A}C$$

Activity 2

Proofs of some Identities:

$$A + \overline{A}B = A + B$$

12b: $A + \overline{A}B = A + AB + \overline{A}B \quad (A + AB = A(B+1) = A) \quad$ (Using 11)

$$= A + B$$

$$AB + \overline{A}C + BC = AB + \overline{A}C$$

$$= AB + \overline{A}C + (A + \overline{A})BC$$

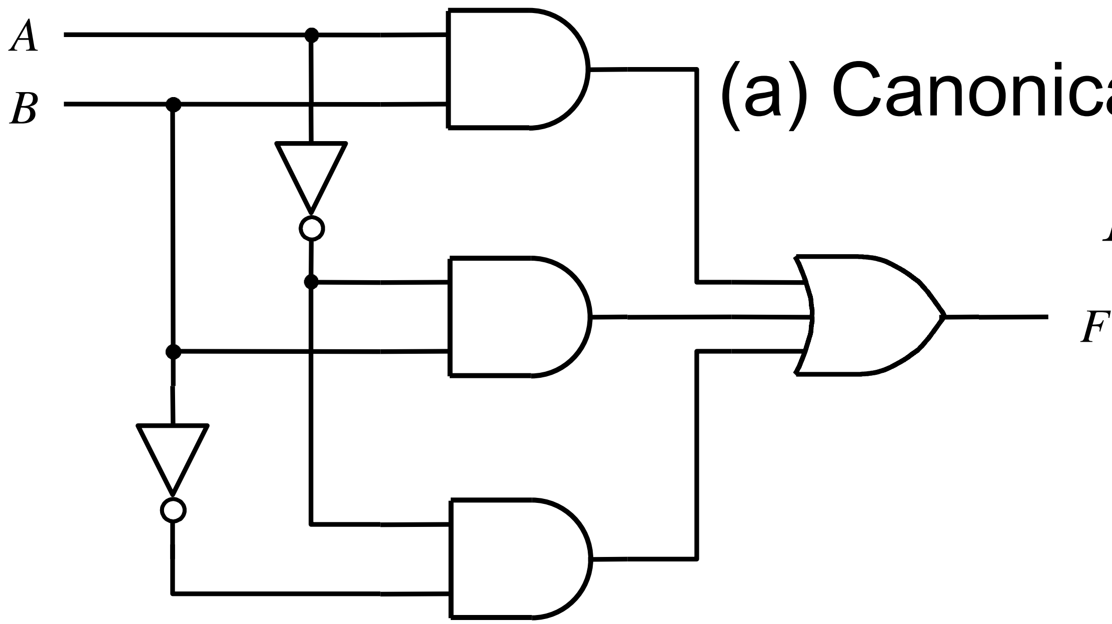
13a: $= AB + \overline{A}C + ABC + \overline{A}BC$

$$= AB(C+1) + \overline{A}C(B+1)$$

$$= AB + \overline{A}C$$

Simplifying Logic Expressions Using Boolean Identities

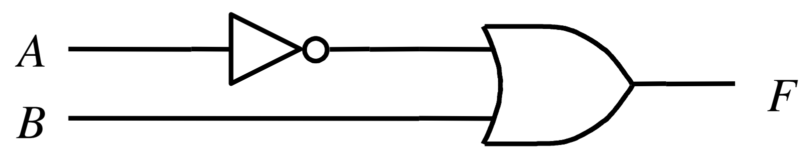
Simplifying Logic Equations – Why?



(a) Canonical sum-of-products

$$F = A.B + \overline{A}.B + \overline{A}.\overline{B}$$

$$F = \overline{A} + B$$



(b) Minimal-cost realization

Simplifying Logic Equations

- Simplifying logic expressions can lead to using smaller number of gates (parts) to implement the logic expression
- Can be done using
  - Boolean Identities (algebraic)
  - Karnaugh Maps (graphical)
- A *minimum SOP* (MSOP) expression is one that has no more AND terms or variables than any other equivalent SOP expression.
- A *minimum POS* (MPOS) expression is one that has no more OR factors or variables than any other equivalent POS expression.
- There may be several MSOPs of an expression

Example of Using Boolean Identities

- Find an MSOP for

$$F = \overline{X}W + Y + \overline{Z}(Y + \overline{X}W)$$

$$= \overline{X}W + Y + \overline{Z}Y + \overline{Z}\,\overline{X}W$$

$$= \overline{X}W(1 + \overline{Z}) + Y(1 + \overline{Z})$$

$$= \overline{X}W + Y$$

Activity 3

- Find an MSOP for

$$F = V\overline{W}XY + VWYZ + V\overline{X}YZ$$

Activity 3

- Find an MSOP for

$$F = V\overline{W}XY + VWYZ + V\overline{X}YZ$$

$$= VY(\overline{W}X + WZ + \overline{X}Z)$$

$$= VY(\overline{W}X + Z(W + \overline{X})) \quad [W + \overline{X} = \overline{\overline{W}X}]$$

$$= VY(\overline{W}X + Z\overline{\overline{W}X}) \quad [A + \overline{A}B = A + B]$$

$$= VY(\overline{W}X + Z)$$

$$= VY\overline{W}X + VYZ$$

Combinational and Sequential Circuits

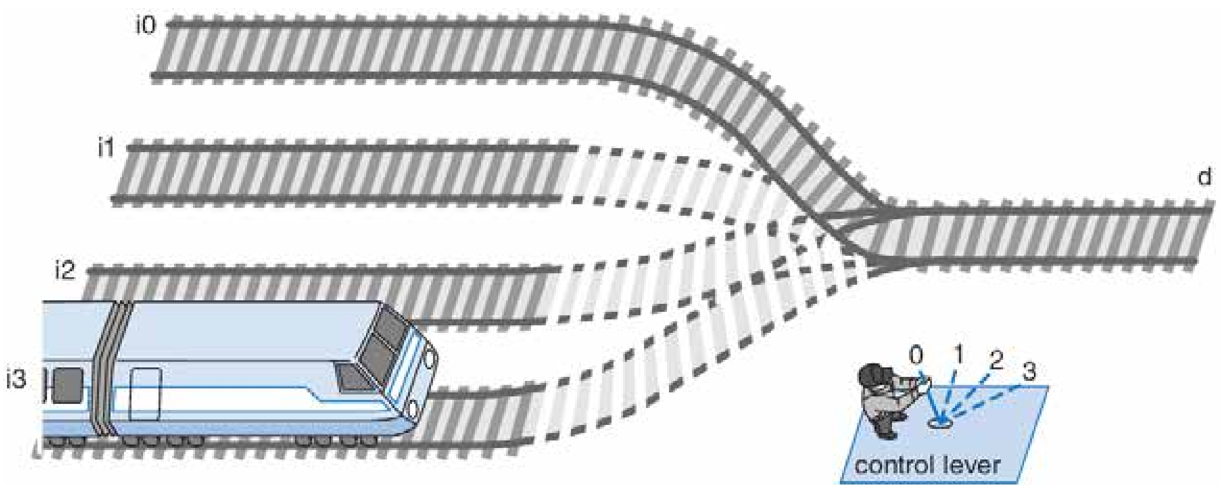
Digital Circuit Classification

- Combinational circuits
  - Output depends only solely on the current combination of circuit inputs
  - Same set of input will always produce the same outputs
  - Consists of AND, OR, NOR, NAND, and NOT gates
- Sequential circuits
  - Output depends on the current inputs and state of the circuit (or past sequence of inputs)
  - Memory elements such as flip-flops and registers are required to store the "state"
  - Same set of input can produce completely different outputs

Combinational Circuits

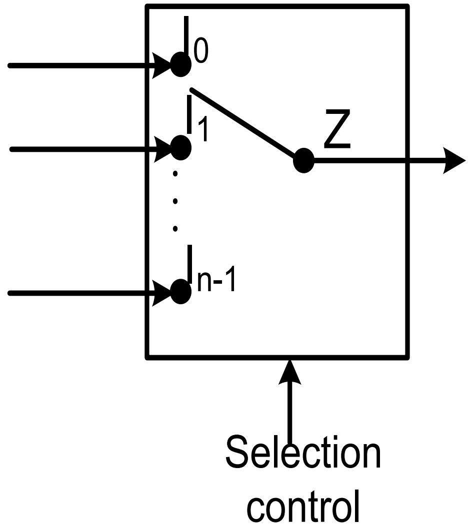
Multiplexer

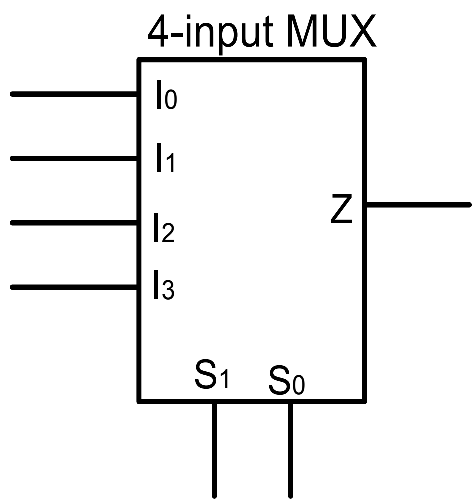
- A multiplexer (MUX) selects data from one of *N* inputs and directs it to a single output, just like a railyard switch
  - 4-input Mux needs 2 select lines to indicate which input to route through
  - N-input Mux needs $\log_2(N)$ selection lines



Multiplexer (2)

- An example of 4-input Mux





Functional block diagram

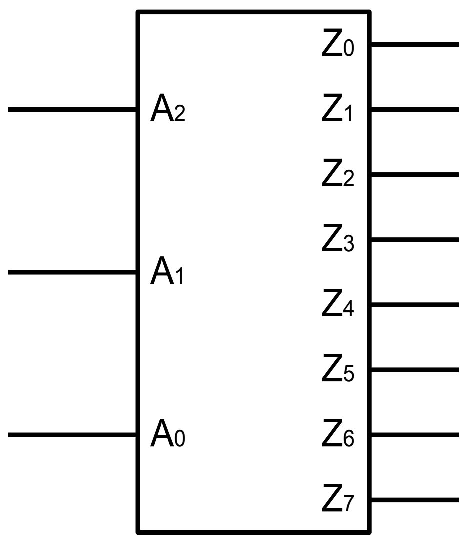
Actual truth table would have 2^6 rows corresponding to I_0, I_1, I_2, I_3, S_0 and S_1

S_1	S_0	Z
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Condensed truth table

Decoder

- A decoder is a circuit element that will decode an *N*-bit code.
- It activates an appropriate output line as a function of the applied *N*-bit input code



Functional block diagram

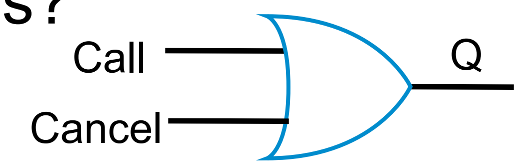
Truth Table

A_2	A_1	A_0	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Sequential Circuits

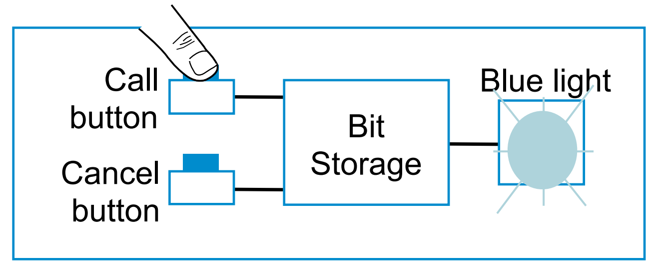
Why Bit Storage ?

- Flight attendant call button
  - Press call: light turns on
    - ***Stays on*** after button released
  - Press cancel: light turns off
  - Logic gate circuit to implement this?

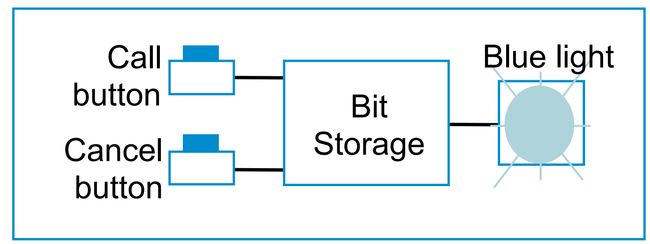


Doesn’t work. Q=1 when Call=1, but doesn’t stay 1 when Call returns to 0

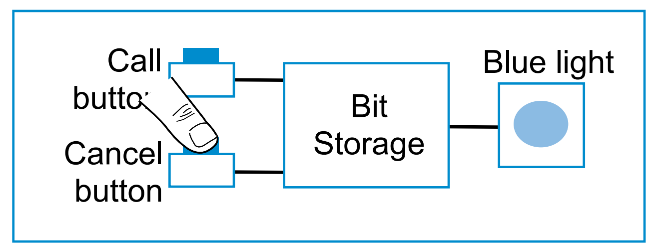
Need some form of “memory” in the circuit



1. Call button pressed – light turns on



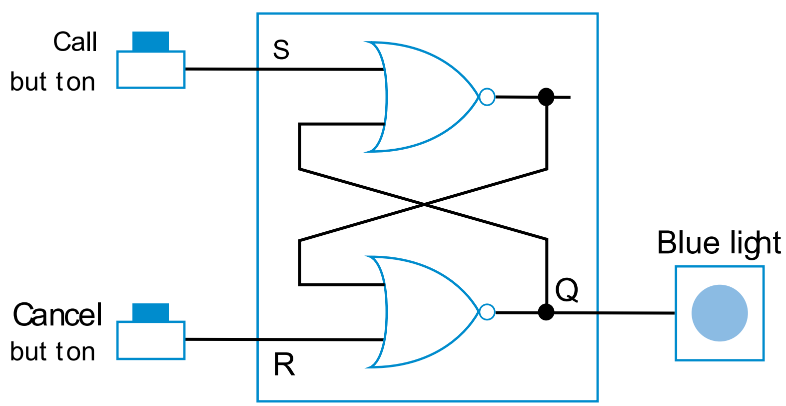
*2. Call button released – light **stays on***



3. Cancel button pressed – light turns off

Bit Storage Using SR Latch

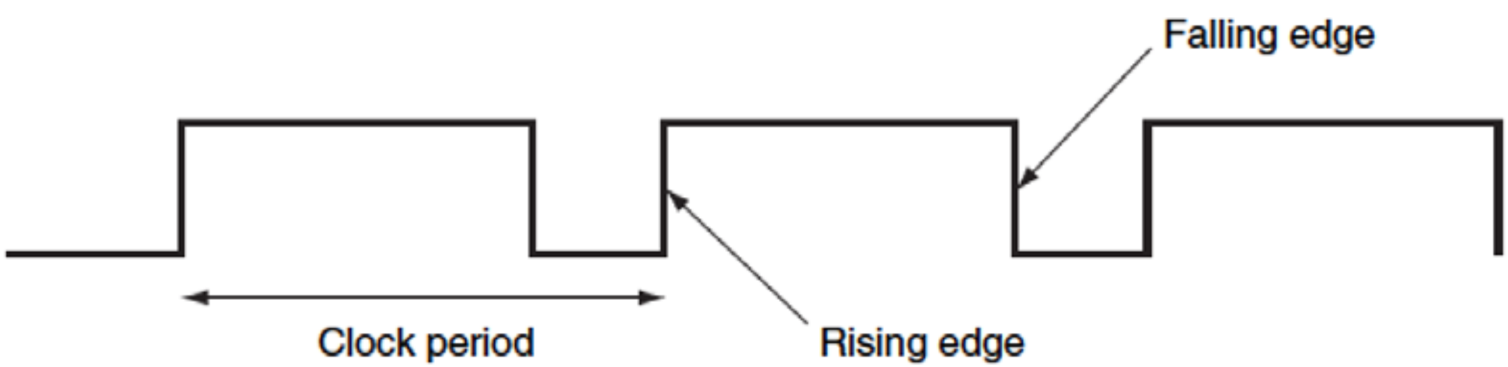
- Simplest memory elements are Latch and Flip-Flops
- SR (set-reset) latch is an ***un-clocked*** latch
  - Output Q=1 when S=1, R=0 (set condition)
  - Output Q=0 when S=0, R=1 (reset condition)
  - Problem - Q is undefined if S=1 and R=1



Clocks

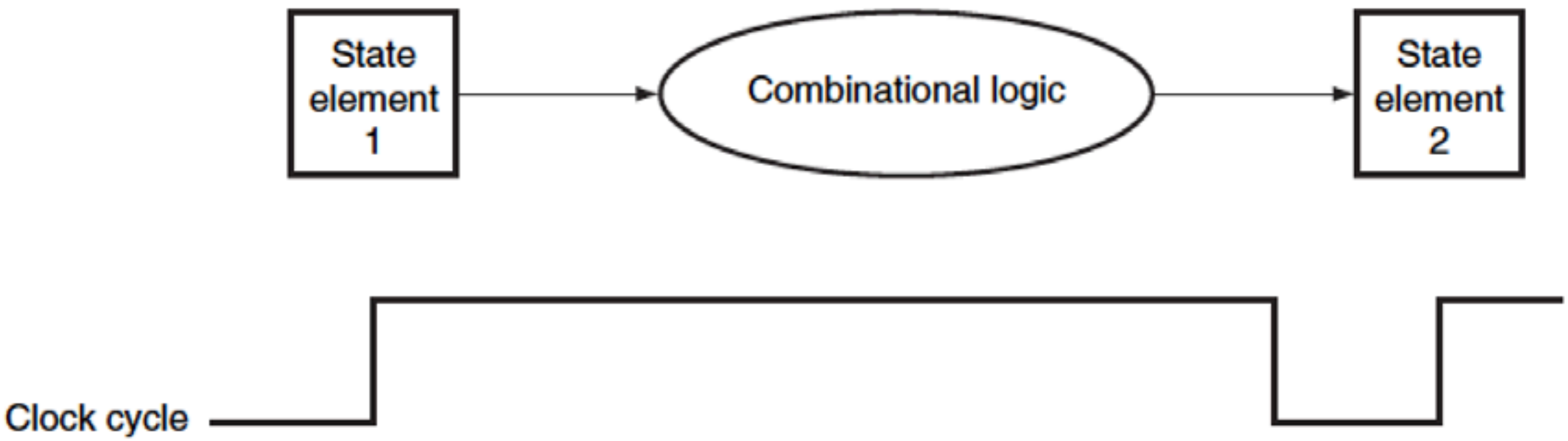
- ***Clock period***: time interval between pulses
  - example: period = 20 ns
- ***Clock frequency***: 1/period
  - example: frequency = 1 / 20 ns = 50 MHz
- ***Edge-triggered clocking***: all state changes occur on a clock edge.

Freq	Period
100 GHz	0.01 ns
10 GHz	0.1 ns
1 GHz	1 ns
100 MHz	10 ns
10 MHz	100 ns



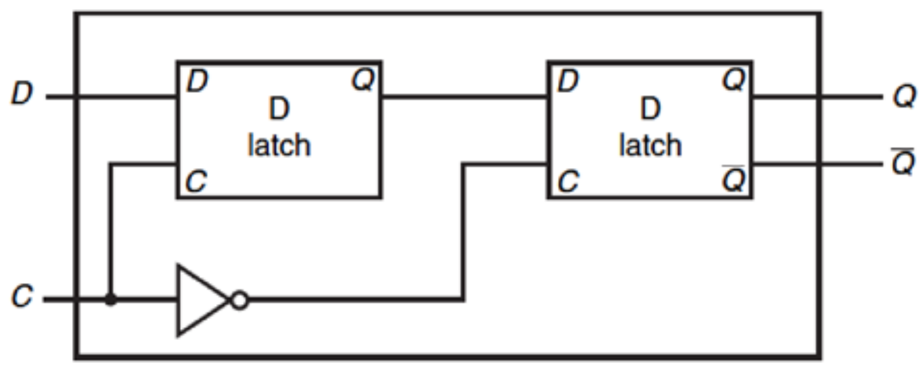
Clock and Change of State

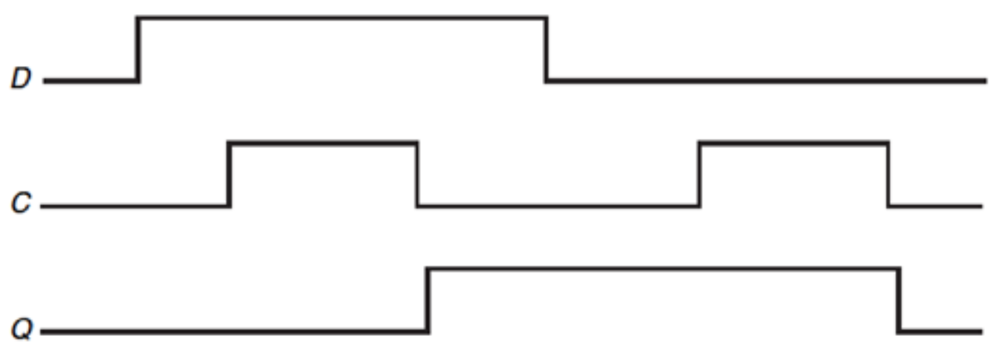
- Clock controls when the state of a memory element changes
- ***Edge-triggered clocking***: all state changes occur on a clock edge.



Clock Edge Triggered Bit Storage

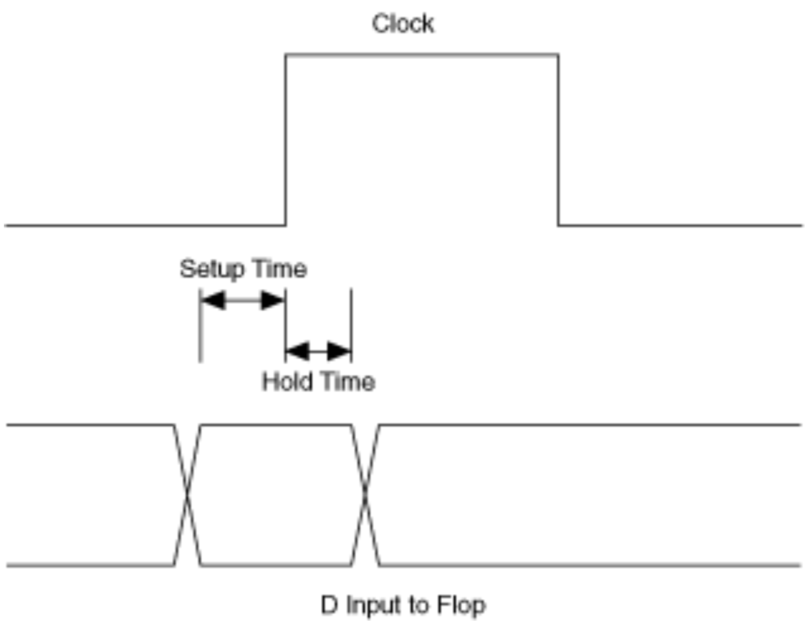
- ***Flip-flop*** - Bit storage that stores on clock edge, not level
- D Flip-flop
  - Two latches, master and slave latches.
  - Output of the first goes to input of second, slave latch has inverted clock signal (falling-edge trigger)





Setup and Hold Time

- Setup time
  - The minimum amount of time the data signal should be held steady before the clock edge arrives.
- Hold time
  - The minimum amount of time the data signal should be held steady after the clock edge.



N-Bit Register

- Cascade *N* number of D flip-flops to form a *N*-bit register
- An example of 8-bit register formed by 8 edge-triggered D flip-flops